Distributed Extended Kalman Filter for Position, Velocity, Time Estimation in Satellite Navigation Receivers

Ondrej JAKUBOV[1,2], *Pavel KOVAR*[1], *Petr KACMARIK*[1], *Frantisek VEJRAZKA*[1]

[1]Dept. of Radio Engineering, CTU in Prague, FEL, Technická 2, 166 27 Prague 6, Czech republic
[2]Nottingham Scientific Limited, Loxley House, Tottle Road, Nottingham, United Kingdom, NG2 1RT

ondrej.jakubov@nsl.eu.com, kovar@fel.cvut.cz

Abstract. *Common techniques for position-velocity-time estimation in satellite navigation, iterative least squares and the extended Kalman filter, involve matrix operations. The matrix inversion and inclusion of a matrix library pose requirements on a computational power and operating platform of the navigation processor. In this paper, we introduce a novel distributed algorithm suitable for implementation in simple parallel processing units each for a tracked satellite. Such a unit performs only scalar sum, subtraction, multiplication, and division. The algorithm can be efficiently implemented in hardware logic. Given the fast position-velocity-time estimator, frequent estimates can foster dynamic performance of a vector tracking receiver. The algorithm has been designed from a factor graph representing the extended Kalman filter by splitting vector nodes into scalar ones resulting in a cyclic graph with few iterations needed. Monte Carlo simulations have been conducted to investigate convergence and accuracy. Simulation case studies for a vector tracking architecture and experimental measurements with a real-time software receiver developed at CTU in Prague were conducted. The algorithm offers compromises in stability, accuracy, and complexity depending on the number of iterations. In scenarios with a large number of tracked satellites, it can outperform the traditional methods at low complexity.*

Keywords

Factor graph, GNSS, Kalman filter, PVT, sum-product algorithm.

1. Introduction

The *global navigation satellite systems* (GNSS) are being developed by many countries nowadays. In 2013, there are more than 60 operational GNSS satellites in orbit. This number is expected to reach 90 within three years. The constantly increasing number of visible space vehicles (SV) and other radio beacons (RB), such as pseudolites and cellular mobile stations, challenges not only the design of a digital signal processor providing standard outputs of *pseudoranges*, *pseudorange rates* and *navigation data* [1–6], but also the design of the position-velocity-time (PVT) fusion algorithm handling large vector data, various coordinate systems and time references. The operations of the PVT estimation/filtering algorithms used in practice involve matrix manipulations whose complexity grows significantly with the increasing number of measurements [7, 27]. These algorithms, including *least squares* (LS), *weighted least squares* (WLS), *extended Kalman Filter* (EKF), strictly rely on first order Taylor linearization of the pseudorange measurement model. The linearization works well for distant SVs and low user dynamics. If distances to narrow RBs are measured or the user maneuvers quickly, the geometry changes rapidly with respect to the time step of PVT estimation and the basic assumptions of the model simplification are violated.

The algorithms such as *unscented Kalman Filter* (UKF), *grid-based filter* (GF), and *particle filter* (PF) can model the nonlinearity based on the representation of the probability density function (PDF) by a finite number of samples [8–12]. These methods are mostly of quadratic or linear dependency on the number of visible RBs, although they work on vector data. However, a large number of particles and frequent resampling are necessary for these algorithms to first converge and second provide estimates with expected precision.

In this study, we focus on facilitating the requirements on the navigation processor. We introduce a novel distributed algorithm suitable for implementation in simple parallel processing units each for a tracked satellite. Such a unit performs only scalar sum, subtraction, multiplication, and division. The algorithm can be efficiently implemented in hardware logic. Given the fast position-velocity-time estimator, frequent estimates can be obtained to generate code and carrier replicas and hence foster dynamic performance of so called vector tracking receiver [13–15].

The algorithm has been designed using *factor graph* (FG) framework and the *sum-product algorithm* (SPA) [16, 17]. First, a FG has been created to intentionally derive the EKF using the SPA relying on known methodology [18]. By splitting vector nodes into scalar ones, we derive a cyclic FG on which we define initialization and iteration routines likewise the SPA in a tree factor graph. A similiar approach

has been adopted by [19, 20] for a one-shot localization of a mobile station in wireless communication systems. Here, we generalize the algorithm and show that it is equivalent to the proposed algorithm when variances of the prediciton messages are set to infinity.

In the study, we further analyse convergence and accuracy using Monte Carlo simulations of the EKF and the proposed method, compare the number of floating point operations, deliver a simulation case study for a vector tracking architecture and experimental measurements with a real-time software receiver developed at CTU in Prague [3].

1.1 Notation

We denote vectors and matrices with bold emphasize throughout the text. All vectors are column vectors. All estimates of values are denoted with hat - an estimate of vector $\mathbf{a}$ is denoted as $\hat{\mathbf{a}}$. Prediction of vector $\mathbf{a}$ is denoted with tilde $\tilde{\mathbf{a}}$. The mean value of random vector $\mathbf{a}$ is denoted as $\mathrm{E}[\mathbf{a}] \triangleq \mu_{\mathbf{a}}$, the covariance matrix as $\mathbf{C}_{\mathbf{a}} \triangleq \mathrm{E}\left[(\mathbf{a}-\mu_{\mathbf{a}})(\mathbf{a}-\mu_{\mathbf{a}})^T\right]$. Diagonal matrices with elements $a_{1,1},\ldots,a_{N,N}$, $N \in \mathbb{N}$ on the main diagonal are denoted as $\mathbf{A} \triangleq \mathrm{diag}(a_{1,1},\ldots,a_{N,N})$. If vector $\mathbf{a}$ represents a measurement in additive noise $\mathbf{w}$, then asterisk denotes the noiseless value $\mathbf{a}^* \triangleq \mathbf{a}-\mathbf{w}$. With notation $\mathcal{N}_{\mathbf{a}}(\mu_{\mathbf{a}}, \mathbf{C}_{\mathbf{a}})$ we mean that random vector $\mathbf{a}$ has multivariate Gaussian (normal) distribution with mean $\mu_{\mathbf{a}}$ and covariance matrix $\mathbf{C}_{\mathbf{a}}$. The first partial derivative of M-dimensional vector function $\mathbf{g}(\theta) = [g_1 \ldots g_M]$ with respect to p-dimensional vector $\theta = [\theta_1 \ldots \theta_p]$, Jacobian matrix, is denoted as follows

$$\frac{\partial \mathbf{g}(\theta)}{\partial \theta} \triangleq \begin{bmatrix} \frac{\partial g_1}{\partial \theta_1} & \cdots & \frac{\partial g_1}{\partial \theta_p} \\ \vdots & \ddots & \vdots \\ \frac{\partial g_M}{\partial \theta_1} & \cdots & \frac{\partial g_M}{\partial \theta_p} \end{bmatrix}. \quad (1)$$

If $g(x_1,\ldots x_K)$ is a scalar function of variables $x_1,\ldots,x_K$, we denote the integral over all the variables except for x_k where $1 \le k \le K$ as

$$\int_{\sim x_k} g(x_1,\ldots,x_K)\,\mathrm{d}x_1 \ldots \mathrm{d}x_K \triangleq \int \ldots \int g(x_1,\ldots,x_K) \underbrace{\mathrm{d}x_1 \ldots \mathrm{d}x_K}_{\text{except for } x_k}. \quad (2)$$

In the factor graph theory, we denote a message from variable node v to factor node F as $\lambda_{v\to F}$, and a message from factor node F to variable node v as $\mu_{F\to v}$.

1.2 Document Structure

In Section 2, we address the problem of PVT estimation in GNSS. In Section 3, we discuss three existing GNSS receiver architectures - *scalar* (conventional), *vector tracking*, and *direct-position estimation* architectures. By definition of measurement equations for each architecture, we show that the derived approach can be applied to both two-step and vector tracking architectures. In Section 5, we overview the theory of factor graphs and the sum-product algorithm. In the next section, an algorithm for Bayesian filtering on the FG with general PDF representation is derived. Using Gaussian PDF representation, update rules for FGs modeling KF, EKF are derived, inspired by [18]. By splitting the vectors into scalars, novel iterative algorithms are introduced.

In Section 9, an explanation to the simulation and experimental philosophy is given as well as a complexity comparison of the employed methods. Section 10 delivers convergence, precision and dynamic analysis using Monte Carlo simulation methods for scalar (conventional) tracking architecture. A case study of incorporation of the proposed method into the vector tracking architecture is presented in Section 11. Experimental results for the scalar (conventional) architecure using a software receiver developed at CTU in Prague are discussed in Section 12.

2. System Model

Conventional GNSS receiver estimates its PVT based on a two-step approach. As the satellites are acquired, tracked by the code and carrier tracking loops, the navigation data are demodulated, a set of pseudoranges $\{\rho_i\}_{i=1}^I$ and pseudorange rates $\{\dot{\rho}_i\}_{i=1}^I$ are formed as the observables for the PVT estimator. Symbol I denotes the number of visible satellites. Pseudorange ρ_i is obtained as the time of reception t_U minus the time of transmission $t_{S,i}$ multiplied by the speed of light c

$$\rho_i = c \cdot (t_U - t_{S,i})$$

with both times measured in the local time scale introducing a bias b between them such that the pseudorange measurement can be modeled as

$$\rho_i = \|\mathbf{x}_U - \mathbf{x}_{S,i}\| + b + w_{\rho,i} \quad (3)$$

where $\mathbf{x}_U = [x\,y\,z]^T$ is the user position at the time reception t_U and $\mathbf{x}_{S,i} = [x_{S,i}\,y_{S,i}\,z_{S,i}]^T$ is the satellite position at the time of transmission $t_{S,i}$, both vectors represented in an earth-centred earth-fixed (ECEF) coordinate frame. Symbol $w_{\rho,i}$ denotes the measurement noise. Pseudorange rate $\dot{\rho}_i$ is obtained as the negative of the measured signal frequency drift $f_{s,i}$ in meters per second

$$\dot{\rho}_i = -\frac{f_{s,i}}{f_c} \cdot c.$$

Symbol f_c denotes the carrier frequency. The pseudorange rate ρ_i is related to the user velocity $\mathbf{v}_U = [\dot{x}_U\,\dot{y}_U\,\dot{z}_U]^T$ by the following equation

$$\dot{\rho}_i = -\mathbf{1}_i^T \cdot (\mathbf{v}_U - \mathbf{v}_{S,i}) + \dot{b} + w_{\dot{\rho},i}$$

where $\mathbf{v}_{S,i} = [\dot{x}_{S,i}\,\dot{y}_{S,i}\,\dot{z}_{S,i}]^T$ is the satellite velocity, $\dot{b}$ is the clock drift, $w_{\dot{\rho},i}$ is the measurement noise and $\mathbf{1}_i$ is the unit

line-of-sight vector between the receiver and the satellite

$$\mathbf{1}_i = \left[\frac{x_U - x_{S,i}}{\|\mathbf{x}_U - \mathbf{x}_{S,i}\|} \; \frac{y_U - y_{S,i}}{\|\mathbf{x}_U - \mathbf{x}_{S,i}\|} \; \frac{z_U - z_{S,i}}{\|\mathbf{x}_U - \mathbf{x}_{S,i}\|} \right]^T.$$

The user velocity $\mathbf{v}_U$ reffers to the time of reception t_U and the satellite velocity $\mathbf{v}_{S,i}$ to the time of transmission $t_{S,i}$. Both vectors are related to an ECEF coordinate frame. Defining the PVT vector $\gamma = \left[\mathbf{x}_U \, b \, \mathbf{v}_U \, \dot{b}\right]$ as a vector of parameters to estimated, the vector of pseudoranges and pseudorange rates $\xi = [\rho_1 \dots \rho_I \dot{\rho}_1 \dots \dot{\rho}_I]^T$ as a measurement vector, the following equation yields an additive noise model for the PVT estimator

$$\xi = \mathbf{g}(\gamma) + \mathbf{w}_\xi \tag{4}$$

where $\mathbf{g}$ is a nonlinear $8 \to 2I$ dimensional function, and $\mathbf{w}_\xi = \left[w_{\rho,1} \dots w_{\rho,I} \, w_{\dot{\rho},1} \dots w_{\dot{\rho},I}\right]$ is an additive noise vector. Provided the system evolves in time, detoning with discrete time index n, all the model quantities and function $\mathbf{g}$ become time dependent

$$\xi_n = \mathbf{g}_n(\gamma_n) + \mathbf{w}_{\xi,n}. \tag{5}$$

Common estimators, such as iterative (W)LS and EKF, linearize the measurement equation, since the geometry changes relatively slowly.

3. Receiver Concepts

The fact that the tracking loops operate decoupled and independently of the PVT estimator is a suboptimal solution. *Vector tracking* concept [14, 15, 21] uses the PVT estimate or prediction to control the tracking channels which increases the tracking sensitivity when some of the satellites are blocked, lowers reacquisition time and increases resistance to interference [22]. The necessity of the high rate operation of the PVT estimator in this case can be suppressed by using *prefilters* of the pseudoranges and pseudorange rates. The PVT estimator can then operate at much lower rate and the control values for the tracking loops are held constant between its successive epochs.

The most advanced *direct positioning* (DPE) approach, so far intractable by current technology in real time, estimates the user PVT directly from the received signal samples. It has been shown that the DPE has similar benefits to the vector tracking concept [23] and is very powerful when multipath mitigation or antenna array processing [24, 25]. The comparison of the GNSS receiver architectures is depicted in Fig. 1.

4. Extended Kalman Filter

Let θ_n be a p-dimensional random vector parameter at time n to be estimated with the following *state-space model*:

$$\theta_n = \mathbf{a}_n(\theta_{n-1}) + \mathbf{B}_n \mathbf{u}_n \tag{6}$$

where $\mathbf{a}_n$ is a p-dimensional function, $\mathbf{u}_n$ is an r-dimensional vector being a Gaussian random variable uncorrelated over time, named as *driving noise*, with zero mean and covariance matrix $\mathbf{Q}_n$

$$\mathrm{E}\left[\mathbf{u}_{n+m}\mathbf{u}_n^T\right] = \begin{cases} \mathbf{0} & n \neq m, \\ \mathbf{Q}_n & n = m \end{cases} \tag{7}$$

and $\mathbf{B}_n$ is a $p \times r$ matrix. Assume the initial value of the parameter is Gaussian $\theta_{-1} \sim \mathcal{N}(\mu_\theta, \mathbf{C}_\theta)$ and independent of $\mathbf{u}_n$ for $n \geq 0$. If $\mathbf{x}_n$ is an $M \times 1$ observation vector expressed by the following additive Gaussian noise model at time n, forming the *measurement equation*,

$$\mathbf{x}_n = \mathbf{h}_n(\theta_n) + \mathbf{w}_n \tag{8}$$

where $\mathbf{h}_n$ denotes an M-dimensional function, $\mathbf{w}_n$ is $M \times 1$ zero mean Gaussian observation vector, $\mathbf{w}_n \sim \mathcal{N}(\mathbf{0}, \mathbf{C}_n)$, uncorrelated over time with covariance matrix $\mathbf{C}_n$

$$\mathrm{E}\left[\mathbf{w}_{n+m}\mathbf{w}_n^T\right] = \begin{cases} \mathbf{0} & n \neq m, \\ \mathbf{C}_n & n = m, \end{cases} \tag{9}$$

the sequential suboptimal MMSE estimator of θ_n based on first order Taylor linearization, *named as extended Kalman filter*, can be summarized by the following recursion [26]:
Prediction:

$$\tilde{\theta}_n = \mathbf{a}_n\left(\hat{\theta}_{n-1}\right). \tag{10}$$

Minimum Prediction MSE Matrix:

$$\tilde{\mathbf{M}}_n = \mathbf{A}_n \mathbf{M}_n \mathbf{A}_n^T + \mathbf{B}_n \mathbf{Q}_n \mathbf{B}_n^T. \tag{11}$$

Kalman Gain Matrix:

$$\mathbf{K}_n = \tilde{\mathbf{M}}_n \mathbf{H}_n^T \left(\mathbf{H}_n \tilde{\mathbf{M}}_n \mathbf{H}_n^T + \mathbf{C}_n\right)^{-1}. \tag{12}$$

Correction:

$$\hat{\theta}_n = \tilde{\theta}_n + \mathbf{K}_n\left(\mathbf{x}_n - \mathbf{h}_n\left(\tilde{\theta}_n\right)\right). \tag{13}$$

Minimum MSE Matrix:

$$\mathbf{M}_n = (\mathbf{I} - \mathbf{K}_n \mathbf{H}_n)\tilde{\mathbf{M}}_n. \tag{14}$$

The recursion is initialized with $\hat{\theta}_{-1} = \mu_\theta$, $\tilde{\mathbf{M}}_{-1} = \mathbf{C}_\theta$. In (11), (12), (14), we substitute

$$\mathbf{A}_n = \left.\frac{\partial \mathbf{a}_n(\theta)}{\partial \theta}\right|_{\hat{\theta}_{n-1}}, \tag{15}$$

$$\mathbf{H}_n = \left.\frac{\partial \mathbf{h}_n(\theta)}{\partial \theta}\right|_{\tilde{\theta}_n}. \tag{16}$$

5. Theory of Factor Graphs and the Sum-Product Algorithm

In this section, we briefly define the factor graph and describe the sum-product algorithm. *Factor graph* (FG) is a bipartite graph that represents relations among variables of a system [16, 17]. The system is assumed to be described by

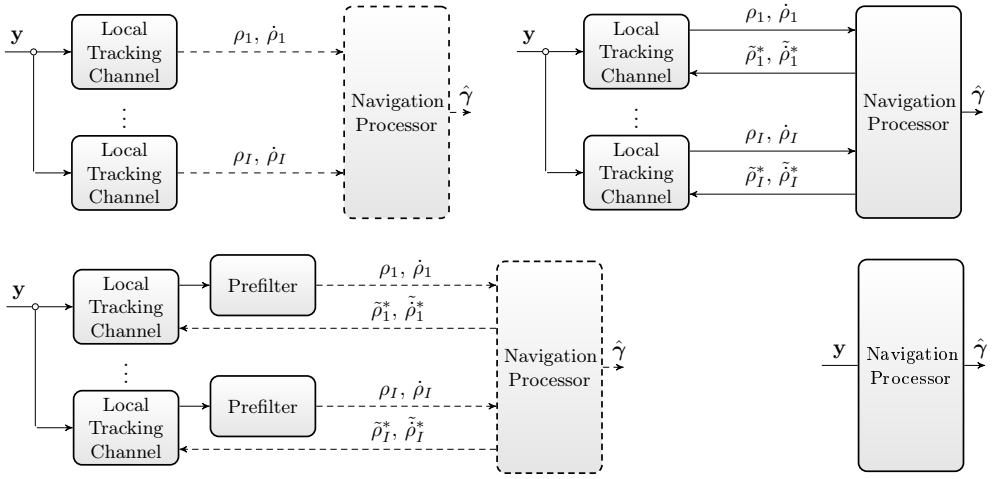

Fig. 1. Receiver architectures: (upper left) conventional architecture, (upper right) vector tracking architecture, (lower left) vector tracking architecture with prefilters, (lower right) direct positioning architecture. Lines denoted as dash operate at low rate (navigation update rate - typically 0.1 s or 1 s), whereas solid lines operate at much higher rate.

a complicated *global function that factors into simpler local functions*, each of which having arguments from a subset of the system variables. The *sum-product algorithm* (SPA) is a generic message-passing (MP) algorithm which operates in a factor graph and *attempts to compute various marginal functions associated with the global function*.

Here, we restrict *factorization and marginalization to multiplication and integration, respectively. The global and local functions will be represented by the probability density functions (PDF)*. Although the factor graph framework applies to more general problems, such constraint will make our approach to PVT filtering more illustrative.

Let's assume that global function $p(X) \geq 0$ of K system variables $X = \{x_1, \ldots, x_K\}$ where $\forall k \in \{1, \ldots, K\} : x_k \in \mathcal{A}_k : \mathcal{A}_k \subset \mathbb{R}$ factors into a product of local functions $\{p_j(X_j) : j \in J \wedge p_j(X_j) \geq 0\}$

$$p(X) = \prod_{j \in J} p_j(X_j)$$

for X_j being a subset of the global function arguments $X_j \subset X$ and j denoting the index of the corresponding local function in set J of such indices. Factor graph is a bipartite graph that visualizes the factorization using three types of components: *variable nodes* - each representing the system variable x_k, *factor nodes* - each representing the local function $p_j(X_j)$, and *edges* - connecting the variable nodes x_k and factor nodes $p_j(X_j)$ if and only if $x_k \in X_j$. An example FG where global function $p(x_1, x_2, x_3, x_4)$ factors into local functions $p_A(x_1, x_2, x_3)$, $p_B(x_3, x_4)$ is in Fig. 2.

The *marginal function* $p_i(x_i)$ is defined as a summation of the global function $p(x_1, \ldots, x_K)$ over all arguments except x_i, denoted by $\sim x_i$,

$$\begin{aligned} p_i(x_i) &\triangleq \int_{\sim x_i} p(x_1, \ldots, x_K) \mathrm{d}x_1 \ldots \mathrm{d}x_K \\ &\triangleq \int_{\sim x_i} p(X) \mathrm{d}X. \end{aligned}$$

In the given example, e.g. the marginal function $p_3(x_3)$ would be computed as

$$p_3(x_3) = \int_{x_1 \in \mathcal{A}_1} \int_{x_2 \in \mathcal{A}_2} \int_{x_4 \in \mathcal{A}_4} p(x_1, x_2, x_3, x_4) \mathrm{d}x_1 \mathrm{d}x_2 \mathrm{d}x_4.$$

The key property that *multiplication distributes over summation is adopted in a cycle-free FG to distributively compute the resulting marginal function* $p_i(x_i)$ *from marginalized local functions* $\int_{\sim x_i} p_j(X_j) \mathrm{d}X_j$, where $\cap_{j \in J} X_j = x_i \vee \emptyset$,

$$\begin{aligned} p_i(x_i) &= \int_{\sim x_i} p(X) \mathrm{d}X \\ &= \int_{\sim x_i} \prod_{j \in J} p_j(X_j) \mathrm{d}X \\ &= \prod_{j \in J} \left(\int_{\sim x_i} p_j(X_j) \mathrm{d}X_j \right). \end{aligned}$$

In the given example

$$\begin{aligned} p_3(x_3) &= \left(\int_{x_1 \in \mathcal{A}_1} \int_{x_2 \in \mathcal{A}_2} p_A(x_1, x_2, x_3) \mathrm{d}x_1 \mathrm{d}x_2 \right) \\ &\cdot \left(\int_{x_4 \in \mathcal{A}_4} p_B(x_3, x_4) \mathrm{d}x_4 \right). \end{aligned} \quad (17)$$

Computation of a *single marginal function in a tree FG* is then a "bottom-up" procedure that starts at the leaf vertices. The leaf vertices send trivial identity functions to their parents which wait for messages from all their children before they start calculation of the local marginal functions. When the marginalization is completed in a vertex, it becomes a child and sends the results to its parents. The algorithm continuous until the target marginal function is obtained.

In our example (Fig. 2), in order to get the resulting marginal function $p_3(x_3)$, variable nodes x_1, x_2 first send simple identity messages $\lambda_{x_1 \to A}(x_1)$, $\lambda_{x_2 \to A}(x_2)$ to factor node A, and so sends the variable node x_4 message $\lambda_{x_4 \to B}(x_4)$

to factor node B. The local marginals, the expressions in parenthesis in (17), are evaluated and information about the results is send in messages $\mu_{A\to x_3}(x_3)$, $\mu_{B\to x_3}(x_3)$ to variable node x_3. According to (17), the resulting marginal function is a product of these local marginals, symbolically,

$$p_3(x_3) = \mu_{A\to x_3}(x_3)\cdot\mu_{B\to x_3}(x_3).$$

We use the term "symbolically", since the massages passed in a FG represent information about a PDF, not necessarily the PDF itself. However, it is instructive to use these messages to denote the corresponding marginal PDFs.

The *sum-product algorithm* is a generalization of the MP algorithm for *efficient calculation of all the marginal functions* associated with the global function. Messages from factor node to variable node are denoted with μ, whereas messages from variable node to factor node with λ.

Let x denote a variable node, F a factor node, and $n(x)$, $n(F)$ the sets of neighbors of variable node x and factor node F, respectively. Symbol p_F denotes a local function corresponding to factor node F. The messages sent in a FG are recursively computed according to the following algorithm [16]:

Variable Node to Factor Node:

$$\lambda_{x\to F}(x) = \prod_{G\in n(x)\setminus\{F\}} \mu_{G\to x}(x),$$

Factor Node to Variable Node:

$$\mu_{F\to x}(x) = \int_{\sim x} p_F(X) \prod_{y\in n(F)\setminus\{x\}} \lambda_{y\to F}(y)\mathrm{d}X$$

where X is a set of all arguments of p_F. The marginalized version of the global function with argument x is named *belief* and is evaluated as

$$B(x) = \prod_{G\in n(x)} \mu_{G\to x}(x).$$

In a tree factor graph, the beliefs truly represent the marginals of the global function. However, when *cycles* appear in the graph, vertices infinitely wait for results from each other. A solution to this problem might be to *initialize the messages and let the sum-product algorithm iterate on the FG*. The convergence is mostly difficult to prove. Nonetheless, "appropriate" initial conditions usually succeed.

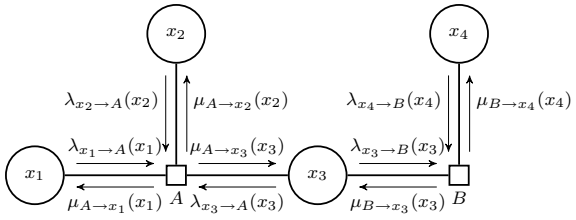

Fig. 2. Example FG with MP for global function $p(x_1,x_2,x_3,x_4)$ that factors into local functions $p_A(x_1,x_2,x_3)$, $p_B(x_3,x_4)$ in the following manner $p(x_1,x_2,x_3,x_4) = p_A(x_1,x_2,x_3)\cdot p_B(x_3,x_4)$. The direction of the messages is denoted with arrow.

6. Distributed Kalman Filter on the FG

Let us assume that θ_n is a random vector parameter to be estimated with the same state-space model and measurement equation as in Section 4, but with both models linear. Using Bayesian rule, posterior PDF $p(\theta_n|\mathbf{x}_{1:n})$ can be decomposed as

$$p(\theta_n|\mathbf{x}_{1:n}) \propto p(\mathbf{x}_n|\theta_n)\int p(\theta_n|\theta_{n-1})\,p(\theta_{n-1}|\mathbf{x}_{1:n-1})\,\mathrm{d}\theta_{n-1} \tag{18}$$

which is depicted in the FG in Fig. 3.

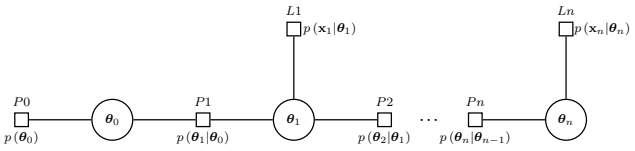

Fig. 3. Factor graph for Bayesian filtering.

The state-space model for the KF can be further decomposed as

$$p(\theta_n|\theta_{n-1}) = \int p(\theta_n|\theta_{n-1},\mathbf{u}_n)\,p(\mathbf{u}_n)\,\mathrm{d}\mathbf{u}_n \tag{19}$$

where

$$p(\theta_n|\theta_{n-1},\mathbf{u}_n) = \prod_{i=1}^{p}\delta\left(\theta_{n,i} - \sum_{j=1}^{p} a_{i,j}\theta_{n-1,j} - \sum_{j=1}^{r} b_{i,j}u_j\right) \tag{20}$$

where $a_{i,j} = [\mathbf{A}]_{i,j}$, $b_{i,j} = [\mathbf{B}]_{i,j}$. Suppose $\mathbf{Q}$ to be diagonal $\mathbf{Q} = \mathrm{diag}\left(\sigma^2_{u_1},\ldots,\sigma^2_{u_r}\right)$ so that

$$p(\mathbf{u}_n) = \prod_{i=1}^{r}\mathcal{N}_{u_i}\left(0,\sigma^2_{u_i}\right). \tag{21}$$

If matrix $\mathbf{Q}$ is not diagonal, *singular value decomposition* (SVD) [26, 27] can be adopted to describe vector $\mathbf{u}$ as

$$\mathbf{u} = \Gamma\eta \tag{22}$$

where η is $r\times 1$ Gaussian vector with zero mean and diagonal covariance matrix, Γ is $r\times r$ matrix that desirably correlates the elements of vector η. Then, we would get

$$p(\theta_n|\theta_{n-1}) = \int p(\theta_n|\theta_{n-1},\eta_n)\,p(\eta_n)\,\mathrm{d}\eta_n \tag{23}$$

where

$$p(\theta_n|\theta_{n-1},\eta_n) = \prod_{i=1}^{p}\delta\left(\theta_{n,i} - \sum_{j=1}^{p} a_{i,j}\theta_{n-1,j} - \sum_{j=1}^{r} c_{i,j}\eta_j\right) \tag{24}$$

where $c_{i,j} = [\mathbf{B}\boldsymbol{\Gamma}]_{i,j}$. Suppose the variance of ith element of vector η is $\sigma^2_{\eta_i}$, then

$$p(\eta_n) = \prod_{i=1}^{r} \mathcal{N}_{\eta_i}\left(0, \sigma^2_{\eta_i}\right). \tag{25}$$

We can decompose the likelihood function in a similiar manner

$$p(\mathbf{x}_n|\theta_n) = \int p(\mathbf{x}_n|\theta_n, \mathbf{w}_n)\, p(\mathbf{w}_n)\, \mathrm{d}\mathbf{w}_n \tag{26}$$

where

$$p(\mathbf{x}_n|\theta_n, \mathbf{w}_n) = \prod_{i=1}^{M} \delta\left(x_{n,i} - \sum_{j=1}^{M} h_{i,j}\theta_{n,j} - w_i\right) \tag{27}$$

where $h_{i,j} = [\mathbf{H}]_{i,j}$. If we suppose that $\mathbf{w}_n$ has diagonal covariance matrix $\mathbf{C}_n = \mathrm{diag}\left(\sigma^2_{w_1}, \ldots, \sigma^2_{w_M}\right)$, then

$$p(\mathbf{w}_n) = \prod_{i=1}^{M} \mathcal{N}_{w_i}\left(0, \sigma^2_{w_i}\right). \tag{28}$$

If $\mathbf{w}_n$ is correlated over elements, the same procedure as for the state-space model can be incorporated to describe $\mathbf{w}_n$ as a product of a matrix and a zero mean Gaussian noise vector with diagonal covariance matrix.

The desired factor graph can be constructed as follows. The vector vertices are split into scalar vertices each of which representing the corresponding vector element. The FG of the state-space model, see Fig. 4 (right), is then constructed based on substituting (20), (21) into (19) resulting in

$$\begin{aligned} p(\theta_n|\theta_{n-1}) = & \\ \int \ldots \int \prod_{i=1}^{p} \delta & \left(\theta_{n,i} - \sum_{j=1}^{p} a_{i,j}\theta_{n-1,j} - \sum_{j=1}^{r} b_{i,j}u_j\right) \\ \cdot \prod_{i=1}^{r} & \mathcal{N}_{u_i}\left(0, \sigma^2_{u_i}\right) \mathrm{d}u_1 \ldots \mathrm{d}u_r. \end{aligned} \tag{29}$$

Similarly, the FG representing the likelihood function, see Fig. 4 (left), is constructed by substituting (27), (28) into (26) resulting in

$$\begin{aligned} p(\mathbf{x}_n|\theta_n) \quad = \quad & \int \ldots \int \prod_{i=1}^{M} \delta\left(x_{n,i} - \sum_{j=1}^{M} h_{i,j}\theta_{n,j} - w_i\right) \\ & \mathcal{N}_{w_i}\left(0, \sigma^2_{w_i}\right) \mathrm{d}w_1 \ldots \mathrm{d}w_M. \end{aligned} \tag{30}$$

The update rules for the factor and variable nodes are depicted in Fig. 5. To derive the update rules for multiple input nodes, we resorted to a recursive evaluation of the update rules for a pair of input nodes in [18].

The *message passing algorithm* starts at variable nodes $\theta_{0,1}, \ldots, \theta_{0,p}$ where the messages sent to factor nodes $P_{1,1}, \ldots, P_{1,p}$ comprise the mean and variance[1] of the initial distribution $\lambda_{\theta_{0,i} \to P_{1,j}}(\theta_{0,i}) = \left\{\mu_{0,i}, \sigma^2_{0,i}\right\}$ for $i, j \in \{1, \ldots, p\}$ where $\mu_{0,i} = [\mu_0]_i$ and $\sigma^2_{0,i} = [\mathbf{C}_0]_{i,i}$. The messages from variable nodes $u_{0,i}$ are also sent to the factor nodes $P_{1,1}, \ldots, P_{1,p}$ $\lambda_{u_{1,i} \to P_{1,j}}(u_{1,i}) = \left\{0, \sigma^2_{u,i}\right\}$ and updates are calculated according to Fig. 5. The branches to variable nodes $\theta_{1,1}, \ldots, \theta_{1,p}$ are disconnected and the resulting messages are sent back towards the variable nodes $\theta_{0,1}, \ldots, \theta_{0,p}$ and the iterations start. When the last iteration, the branches to the variable nodes $\theta_{1,1}, \ldots, \theta_{1,p}$ are connected, the messages are sent to them. The observed values are sent to the likelihood factor nodes $\lambda_{x_{n,i} \to L_{n,i}}(x_{n,i}) = \{x_{n,i}, 0\}$ for $i \in \{1, \ldots, M\}$ and so do the noise variable nodes $\lambda_{w_{n,i} \to L_{n,i}}(w_{n,i}) = \left\{0, \sigma^2_i\right\}$. Next, the messages from the previous state-space factor nodes $P_{1,1}, \ldots, P_{1,p}$ are sent through variable nodes $\theta_{1,1}, \ldots, \theta_{1,p}$ to the likelihood factor nodes $L_{1,1}, \ldots, L_{1,p}$ and the updates are therein calculated. The results are then sent back to variable nodes $\theta_{1,1}, \ldots, \theta_{1,p}$ and then updated taking the unchanged messages from factor nodes $P_{0,1}, \ldots, P_{0,p}$. The branches to the future factor nodes $P_{2,1}, \ldots, P_{2,p}$ remain disconnected. The updated messages are sent towards factor nodes $L_{1,1}, \ldots, L_{1,p}$ and iterations are started in this way. The iterations are finished after the last updates at variable nodes $\theta_{1,1}, \ldots, \theta_{1,p}$ for which the future variable nodes $P_{2,1}, \ldots, P_{2,p}$ are now connected. The messages sent to these factor nodes represent the beliefs and the means can be used to obtain the approximated MMSE estimates. And so continuous the algorithm sequentially.

7. Distributed Extended Kalman Filter on the FG

The *extended Kalman filter* can be also approximated on the scalar FG. Suppose that

$$\mathbf{a}(\theta_n) = \begin{bmatrix} a_1(\theta_{n,1}, \ldots, \theta_{n,p}) \\ \vdots \\ a_p(\theta_{n,1}, \ldots, \theta_{n,p}) \end{bmatrix} \tag{31}$$

and

$$\mathbf{h}(\theta_n) = \begin{bmatrix} h_1(\theta_{n,1}, \ldots, \theta_{n,p}) \\ \vdots \\ h_M(\theta_{n,1}, \ldots, \theta_{n,p}) \end{bmatrix}. \tag{32}$$

Let us next assume that $h_{i,j} = [\mathbf{H}]_{i,j}$ and $a_{i,j} = [\mathbf{A}]_{i,j}$. The conditional PDFs $p(\theta_n|\theta_{n-1}, \mathbf{u}_n)$, $p(\mathbf{x}_n|\theta_n, \mathbf{w}_n)$ both can be approximated for the EKF as Gaussian

$$\begin{aligned} & p(\theta_n|\theta_{n-1}, \mathbf{u}_n) = \\ & \prod_{i=1}^{p} \delta\left(\theta_{n,i} - a_i(\theta_{n-1,1}, \ldots, \theta_{n-1,p}) - \sum_{j=1}^{r} b_{i,j}u_j\right), \end{aligned} \tag{33}$$

$$p(\mathbf{x}_n|\theta_n, \mathbf{w}_n) = \prod_{i=1}^{M} \delta\left(x_{n,i} - h_i(\theta_{n,1}, \ldots, \theta_{n,p}) - w_i\right). \tag{34}$$

The update rules of the scalar variable node remain unchanged compared to the KF's. However, the mean of the factor node update will change and the variance will remain

[1] The initial cross-correlation between the parameter is not assumed. To regard it, one can resort to SVD decomposition.

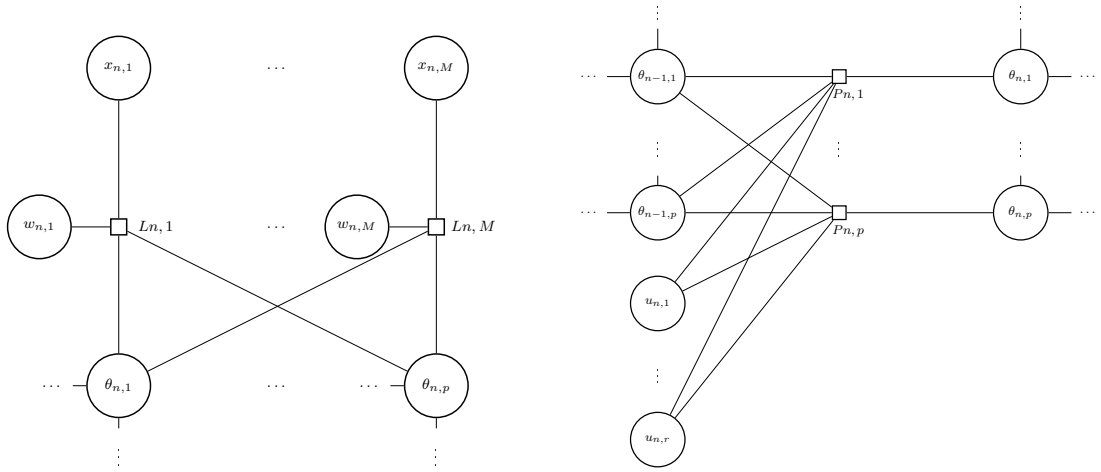

Fig. 4. FG for the likelihood function (left), state-space model of scalar KF (right).

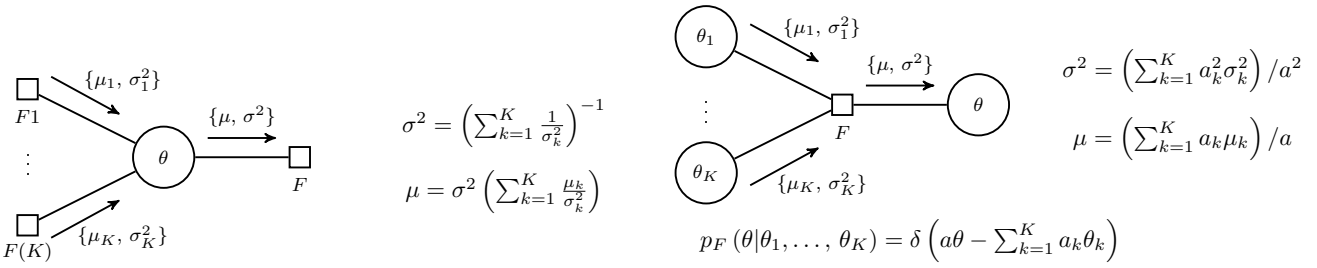

Fig. 5. Update rules for scalar vertices of KF - (left) variable node, (right) factor node.

unchanged. Assume the situation in Fig. 5 (right). Next, assume the PDF (= factor function) $p_F(\theta|\theta_1,\ldots,\theta_K)$ will have the following form

$$\begin{aligned} p_F(\theta|\theta_1,\ldots,\theta_K) &= \delta(a\theta - f(\theta_1,\ldots,\theta_K)) \qquad (35)\\ &\approx \delta\left(a(\theta-\breve{\theta}) - \sum_{k=1}^{K} a_k(\theta_k - \breve{\theta}_k)\right) \end{aligned}$$

where f is K-dimensional function and $\breve{\theta}, \breve{\theta}_1,\ldots,\breve{\theta}_K$ are constant linearizing points such that

$$\breve{\theta} = f(\breve{\theta}_1,\ldots,\breve{\theta}_K). \qquad (36)$$

The updated mean will have the following form

$$\mu = \breve{\theta} + \left(\sum_{k=1}^{K} a_k(\mu_k - \breve{\theta}_k)\right)/a. \qquad (37)$$

In the scalar EKF, the linearization will generally take place in both state-space and likelihood factor nodes. The linearizing points for the state-space model nodes will be the estimated parameters from the previous time $\hat{\theta}_{n-1,i}$ where $i \in \{1,\ldots,p\}$. The linearizing points for the likelihood nodes will be the means of the output messages from the state-space model nodes. This complies with the vector EKF where predictions are used to linearize the nonlinear observation function **h**.

The complexity of the scalar EKF is slightly higher than that of the scalar KF due to the necessity of evaluation of the Jacobians **A**, **H** if they depend on time. The number of flops might be increased if functions **a**, **h** are complicated. The linearizing point can be updated after every iteration to improve the convergence which requires the evaluations of the Jacobians.

8. Complexity Comparison

The advantage of the scalar Kalman filter is that the update rules operate on scalars and simple arithmetic operations unlike the vector case. The complexity of the algorithm is quadratic in the number of states and linear in the number of observations. This is not true for the vector KF where the complexity is cubic in states and cubic or quadratic in observations depending on the form of the KF. However, the sum-product algorithm in the scalar case does not yield truly the MMSE estimates of the parameter θ_n, since the cycles are present in the graph. The complexity then will be a multiple of the number of iterations and the accuracy with convergence will have to be investigated for every single implementation of the scalar Kalman filter.

The *number of flops* required to calculate a recursion is

$$\begin{aligned} O_{\text{sKF}} &= N_{\text{Iter., Ss.}}\left(5p^2+5pr+p\right)\\ &+ \left(N_{\text{Iter., Ss.}}-1\right)\left(4p^2+4pr\right)\\ &+ N_{\text{Iter., Lh.}}\left(9pM+6M\right) \qquad (38) \end{aligned}$$

where p is the size of the state-space vector θ_n, M is the size of the observation vector $\mathbf{x}_n$, r is the size of the driv-

ing noise vector $\mathbf{u}_n$, $N_{\text{Iter., Lh.}}$ is the number of the iterations among the likelihood factor nodes and the state variable nodes, $N_{\text{Iter., Ss.}}$ is the number of the iterations among the state factor nodes and the state variable nodes. In Fig. 6, the number of flops depending on the number of visible SVs is depicted for the EKF and the first three iterations on the scalar EKF ($N_{\text{Iter., Ss.}} = 1$). The EKF is assumed to be in information form [26] which changes the calculation of the inverse in 12 into a quadratic-dependent form on I using matrix inversion lemma. Note that the number of flops is lower for the scalar filter with a single iteration, for two iterations both curves are close to each other, and for three iterations the EKF has lower number of flops.

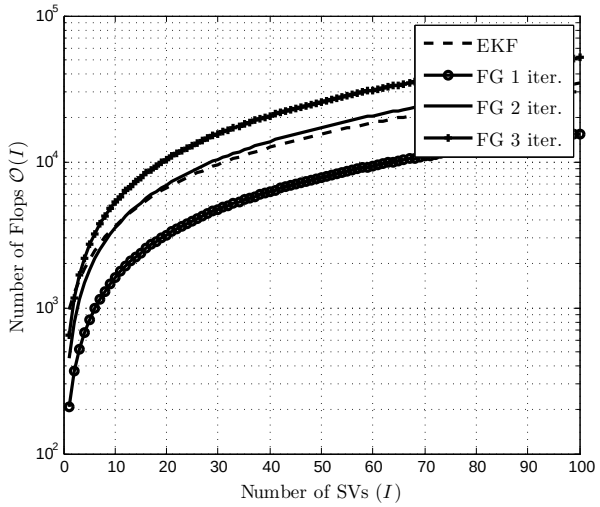

Fig. 6. Complexity comparison of the PVT filtering algorithms. Symbol $O(I)$ denotes the number of floating point operations (flops) depending on the number of SVs denoted as I.

9. Simulations and Experiments

In this section, we investigate convergence and accuracy of the EKF and the proposed algorithm. In practice, the user sometimes manueveres with higher dynamics than reflected by the model, therefore we also investigate the cases where the state-space model is designed for much lower dynamics than the user moves with.

During the simulation results, we will notice that a single iteration can offer similar accuracy to the EKF in low dynamic scenarios, but the performance improves with iterations as the dynamics increase and the EKF can be outperformed. Hence, the FG-based filter offers compromises in accuracy and complexity.

Firstly, we deliver a simple simulation where the convergence and accuracy in a scalar tracking architecture is investigated using Monte Carlo methods, being the subject of Section 10. In Section 11, the filters are incorporated into a vector tracking architecture. The former simulation assumes Gaussian uncorrelated measurements of pseudoranges and pseudorange rate, whereas the latter produces measurements that are correlated and biased by randomly generated values depending on the satellite elevations. Section 12 presents few case studies of the experiment with a real-time receiver - WNav [3], working in the scalar tracking architecture.

10. Simulation - Scalar Tracking Architecture

In this simulation, we investigate convergence and accuracy of the EKF and the FG-based scalar iterative EKF algorithm. The model we adopt is basic, see next section. The pseudoranges and pseudorange rates are Gaussian and uncorrelated without any biases. The goal is to evaluate the performance characteristics in a simple manner and over a large number of realizations. In the first subsection, we introduce the simulation scenario, in the next two subsections the convergence and accuracy results are discussed and finally summarized in the last subsection.

10.1 Simulation Scenarios

We consider from 16 to 64 visible SVs randomly distributed over the open sky, scenarios with low dynamics ($a = 0.1$ m/s^2) and scenarios with moderate dynamics ($a = 1$ m/s^2). The user moves in a circle on the surface of the Earth with radius 1 km, velocity 10 m/s for low dynamics and 33 m/s for moderate dynamics. The estimates were updated every ten times per second $T_N = 0.1$ s.

The standard deviation of the generated pseudorange measurements was $\sigma_\rho = 1$ m, and $\sigma_{\dot\rho} = 0.05$ m/s of the pseudorange rate measurements. These quantities were generated as uncorrelated Gaussian random variables with mean equal to the noiseless values with respect to the geometry and clock offsets. The standard deviation of the elements of the driving noise vector was assumed constant - either $\sigma_u = 0.01$ m/s for high filtering or $\sigma_u = 0.1$ m/s for low filtering. We repeated each simulation 1000 times and simulated over 1000 s. The convergence and accuracy were examined for various number of iterations $N_{\text{Iter.}} \triangleq N_{\text{Iter. Lh.}}$, $N_{\text{Iter. Ss.}} = 1$.

To evaluate the performance characteristics, we define the total position error (TPE) of the position and clock bias estimation as

$$\text{TPE} = \frac{1}{NL}\sum_{n=1}^{N}\sum_{l=1}^{L}\sqrt{\left\|\hat{\mathbf{x}}_{U,n}^{(l)} - \mathbf{x}_{U,n}^{(l)}\right\|^2 + (\hat{b}_n^{(l)} - b_n^{(l)})^2} \quad (39)$$

where n is the time index, N is the number of observations in time, l is the index of the realization, L is the number of realizations. Similarly, we define the total velocity error (TVE) as

$$\text{TVE} = \frac{1}{NL}\sum_{n=1}^{N}\sum_{l=1}^{L}\sqrt{\left\|\hat{\mathbf{v}}_{U,n}^{(l)} - \mathbf{v}_{U,n}^{(l)}\right\|^2 + (\hat{\dot{b}}_n^{(l)} - \dot{b}_n^{(l)})^2}. \quad (40)$$

We claim that the divergence is reached if TPE>100 m or TVE>2 m/s.

10.2 Convergence

The convergence histograms of the EKF and FG-based scalar iterative EKF algorithms are depicted in Fig. 7. In Fig. 7(a), the driving noise std. equals the change of the velocity vector in magnitude over an epoch. In this case, we can infer that the convergence probability is strictly dependent on the number of visible SVs and is always worse for the FG-based EKF filter than for the standard EKF filter. In Fig. 7(b), the same scenario is assumed for a larger number of SVs. For the number of visible SVs equal to sixteen and more, no divergence has been observed in 1000 repetitions. This fact substantiates us to employ the iterative filtering for large data vectors. Luckily, we see that the convergence is relatively fast and can be reached within few iterations.

In the experiment in Fig. 7(c), we increase the user velocity and produce the user acceleration of 1 m/s. The driving noise is left unchanged so that the velocity changes approximately ten times faster than the state-space model assumes. We observe that the FG-based filter and the EKF can still withstand such unmodeled dynamics and converge similarly as in the previous case. However, this is not the case in Fig. 7(d) where the driving noise variance is increased to 0.1 m/s to account for the change in velocity. The lack of averaging of the iterative algorithm causes divergence even for a large number of SVs of the FG filter, the EKF remains stable. We deduce that if the dynamics increase significantly, we had better not model it. If the modeled dynamics is acceptably low at high dynamics, iterations generally improve convergence probability.

10.3 Accuracy

In Fig. 9, we illustrate the accuracy of the algorithms. The TPEs are in the left column, whereas the TVEs are in the right column. Each row corresponds to the same scenario. In Fig. 9(a-b), the same low dynamics matched model is considered. In this case, the FG-based filter outperforms the EKF if the number of SVs is larger than 16 and even for a single iteration. The explanation for this surprising fact is that the EKF is a suboptimal approximation of the KF based on first order Taylor linearization. The FG-based filter linearizes the variables in a scalar, but different, manner and the performance of both filters may generally differ depending on different factors.

If we increase the acceleration as in Fig. 9(c-d), the FG-based filter outperforms the EKF in accuracy for larger number of SVs or for more than one iteration. In this case, the performance is improved over iterations due to the fact that a linearizing point is calculated for each iteration. The linearization fails for a large position difference between the predicted (also linearizing) value and the true value. Here, we iteratively shift towards the true value with iterations, whereas the EKF does this only once at an epoch.

If we model the higher dynamics and remain stable as in Fig. 9(e-f), the EKF performs better and the iterations do not improve the accuracy, but the difference in accuracy between the two filters decays with larger number of the SVs.

10.4 Summary

To summarize the simulation results, we recall that one should use the FG-based filter for PVT estimation if and only if the number of visible SVs is larger than 16 to ensure the convergence. If the user is expected to move from time to time with higher dynamics, it is better not to model it and rather increase the number of iterations. The accuracy can be improved in this case compared to the EKF.

11. Simulation - Vector Tracking Architecture

In this section, we demonstrate the functionality of the proposed algorithm in the vector tracking architecture. We present a case study where we compare the scalar tracking architecture using the EKF with the vector tracking architecture with the EKF or the FG-based scalar iterative EKF. This time correlation among the pseudoranges will be introduced and the biases in the measurement caused by atmospheric effects will be added. We demonstrate that the FG-based filter can withstand these anomalies even in the vector tracking architecture. The first subsection shortly explains the employed simulation methodology. In the second subsection, the simulation results are discussed and the last subsection highlights the key implementation aspects.

11.1 Simulation Methodology

The simulation setup is documented in Tab. 2. Two user scenarios were investigated: low dynamics (LD) scenario, and moderate dynamics (MD) scenario. In either scenario, the user moves in a circle with constant circular orbit speed. The TPE and TVE were calculated over time. For the first part of the simulation, we assume that the user is in an open-sky scenario, whereas in the second part C/N_0 of all visible SVs is swept from 50 dB-Hz down to 10 dB-Hz. Random atmospheric delay errors were artificially added to the propagation times with distribution depending on the elevation angle of the SV. Example values of these errors for the simulation in Fig. 9(a,b) are in Tab. 1. Oscillator phase noise was added to the code delay and the carrier phase. Finite bandwidth of 10 MHz was introduced. The phase noise values $\mathcal{L}(f_0)$, $\mathcal{L}(f_1)$ at frequencies f_0, f_1, respectively, for the carrier phase are in Tab. 2. No multipath, interference, jamming and fading effects were examined.

The simulation complexity has been reduced by avoiding the bit-true multiplication and accumulation (MAC) between the received IF signal samples and the generated replicas. The signals at the output branches of the correlators were generated instead [28–30].

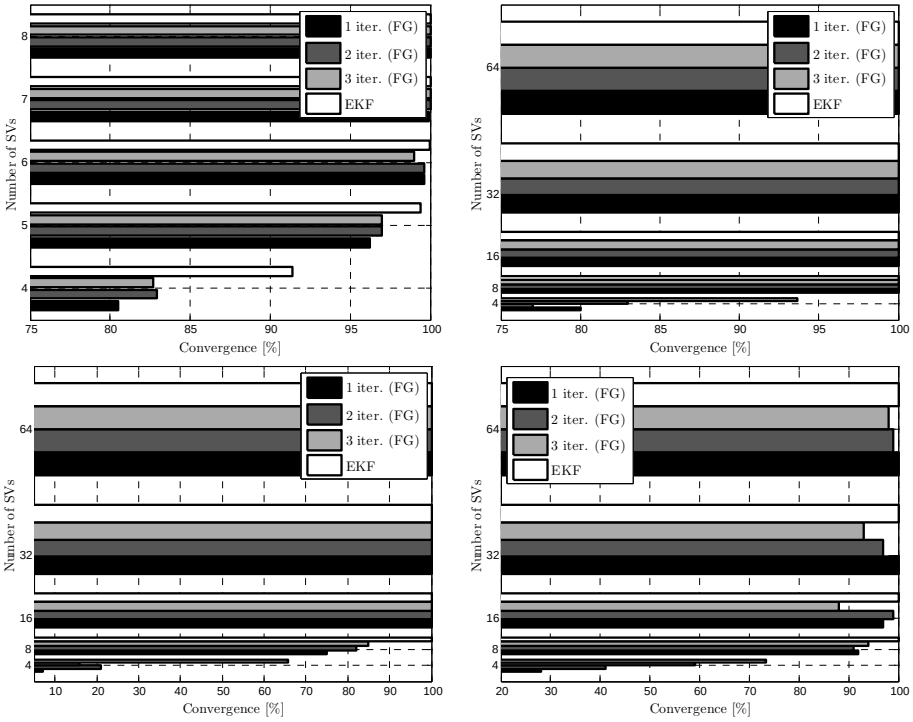

Fig. 7. Simulation results - convergence. (a - upper left) $a = 0.1\,\mathrm{m/s^2}$, $\sigma_u = 0.01\,\mathrm{m/s}$ detail on low number of visible SVs, (b - upper right) $a = 0.1\,\mathrm{m/s^2}$, $\sigma_{u_u} = 0.01\,\mathrm{m/s}$, (c - lower left) $a = 1\,\mathrm{m/s^2}$, $\sigma_u = 0.01\,\mathrm{m/s}$, (d - lower right) $a = 1\,\mathrm{m/s^2}$, $\sigma_u = 0.1\,\mathrm{m/s}$.

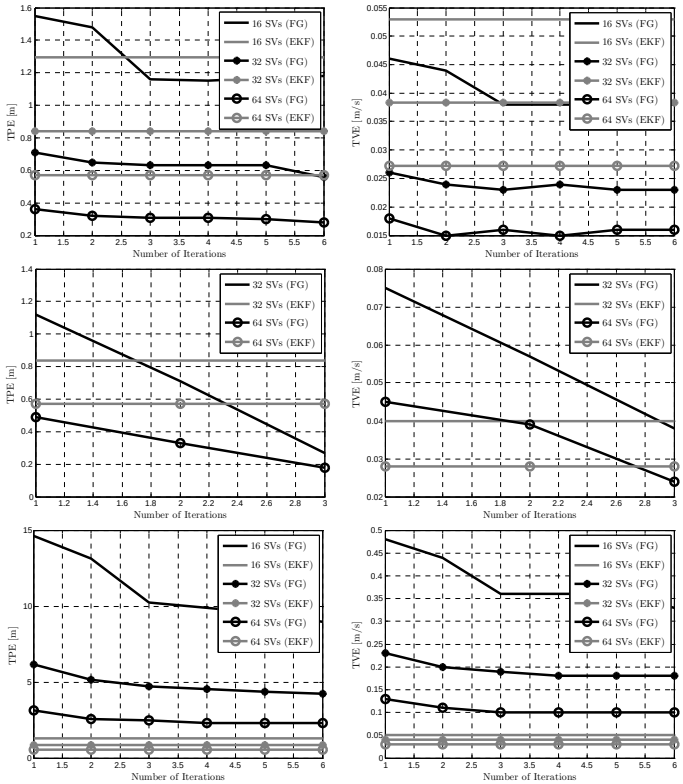

Fig. 8. Simulation results - accuracy. Total position errors (TPE) are in the left column, total velocity errors (TVE) are in the right column. (a - upper left), (b - upper right) $a = 0.1\,\mathrm{m/s^2}$, $\sigma_u = 0.1\,\mathrm{m/s}$, (c - middle left), (d - middle right) $a = 1\,\mathrm{m/s^2}$, $\sigma_u = 0.01\,\mathrm{m/s}$, (e - lower left), (f - lower right) $a = 1\,\mathrm{m/s^2}$, $\sigma_u = 0.1\,\mathrm{m/s}$.

El.	32	42	9	10	21	6	33	40	6
AE	1.1	-0.3	-3.7	0.9	-0.1	-2.3	1.3	-2.0	-2.5

Tab. 1. Examples of atmospheric errors (AE) [m] with elevations of the SVs (El.) [°], for simulation in Figs. 9 (a, b).

Testing Signal	GPS L1 C/A
Predetection Integration Time	$N_b N_c T_c = 20$ ms
Navigation Update Time	$T_N = 0.1$ s
IF Filter Bandwidth	$BW = 10$ MHz
Code Delay Detector	Normalized Power
Carrier Phase Detector	Atan2
C/N_0 Estimator	Squaring [15]
Prefilter	Sequential WLS
User Velocity, Radius, Accel. (LD)	2 m/s, 100 m, 0.04 G
Velocity Driv. Noise Std. (LD)	$\sigma_{\dot{x}} = \sigma_{\dot{y}} = \sigma_{\dot{z}} = 1$ m/s
Clk. Drift Driv. Noise Std. (LD)	$\sigma_{\dot{b}} = 10^{-2}$ m/s
User Velocity, Radius, Accel. (MD)	30 m/s, 100 m, 1 G
Velocity Driv. Noise Std. (MD)	$\sigma_{\dot{x}} = \sigma_{\dot{y}} = \sigma_{\dot{z}} = 20$ m/s
Clk. Drift Driv. Noise Std. (MD)	$\sigma_{\dot{b}} = 10^{-3}$ m/s
Number of Visible SVs	$I = 9$
Init. Lattitude, Longitude, Altitude	0°, 0°, 0 m
Phase Noise	$\mathcal{L}(1\,\text{Hz}) = -30$ dBc/Hz
	$\mathcal{L}(10\,\text{Hz}) = -50$ dBc/Hz
DLL (Order, Bandwidth)	1. order, $B_n = 0.1$ Hz
FLL (Order, Bandwidth)	2. order, $B_n = 10$ Hz

Tab. 2. Simulation setup for the vector tracking architecture.

11.2 Results

The simulation results documenting the open-sky scenario are in Figs. 9, 10. Figs. 9(a-d) depict the situation for LD, whereas Figs. 10(a-d) depict MD. TPE and TVE are plotted for a single iteration ($N_{\text{Iter.}} = 1$) and for three iterations ($N_{\text{Iter.}} = 3$). The proposed FG-based algorithm incorporated to the vector tracking architecture (FG-VDLL/FG-VFLL) is compared with the EKF of the scalar tracking architecture (DLL/FLL) and with the EKF of the vector tracking architecture (VDLL/VFLL). All algorithms adopt identical motion model. Second order FLL aids first order DLL in the scalar tracking loops.

It is clear from Figs. 9(a,b), 10(a,b) that a single iteration on the FG results in comparable position and velocity filtering errors as for the EKF vector tracking loop in an open-sky scenario at low dynamics. An increased number of iterations slightly improves the performance at low dynamics, compare Figs. 9(a,c) and Figs. 9 (b,d). At moderate dynamics, both VDLL/VFLL and FG-VDLL/FG-VFLL perform similarly for $N_{\text{Iter.}} = 3$, see Fig. 10(c,d). For $N_{\text{Iter.}} = 1$, conclusions about precision are difficult to make, see Figs. 10(a,b). The reason is that our motion model does not regard user acceleration. However, the figures document that the proposed method might be able to withstand such unmodeled anomalies, likewise the EKF method. The improper choice of the motion model here causes that the scalar architecture outperforms the vector one for velocity estimation at moderate dynamics, see Figs. 10(b,d). This could be claimed as a disadvantage of the vector tracking architecture.

In Figs. 11(a,b), we sweep C/N_0 for all the tracked SVs, see Fig. 11(c), and observe the stability of the proposed FG-based algorithm in comparison with the EKF-based one. Moderate dynamics and a single iteration are considered. It is apparent from the figures that the FG-VDLL/FG-VFLL loop looses lock at approximately 2 dB lower C/N_0 than the VDLL/VFLL loop. Fig. 11(d) depicts the C/N_0 estimate errors of the FG-VDLL/FG-VFLL channels.

11.3 Implementation Aspects

Likewise for the EKF architecture, it is clear that precision and stability of the loop is highly dependent on the choice of the motion model, predetection integration time and the navigation update time. When a proper motion model is selected, the performance can be improved with increased number of iterations on the FG. But with low number of iterations, the navigation processor might be able to operate at higher update rate which would foster the overall performance.

In our simulation, we did not model any delay between the SV channels and the navigation processor. It represents the situation of a SDR receiver (ipexSR [2], WNav - only PC processing [3]). On the other, the performance might be worse for traditional receiver architectures (WNav - local tracking channels in FPGA, the navigation processor in PC [3]) where such a delay occurs.

12. Experiment - Scalar Tracking Architecture

In this section, we demonstrate the functionality of the proposed FG-based scalar iterative EKF algorithm in a scalar tracking architecture using a GNSS receiver developed at CTU in Prague [3] and high-fidelity GPS L1 Spirent simulator GSS6560. The first subsection explains the employed methodology, next the results are discussed.

12.1 Methodology

We assume that the user moves in a circle with constant circular orbit speed in an open-sky scenario. Firstly, we place the user to 14 static points on the Earth, see Fig. 13, at a given time and investigate the filtering performance. Secondly, two dynamic scenarios ($a = 1$ m/s^2, $a = 10$ m/s^2) with radius 10 km, 1 km and velocity 100 m/s are supposed. The number of visible SVs has always been 11. The measurements were taken for 1 hour. The navigation update time was $T_N = 0.1$ s. Second order DLL, PLL were used with equivalent loop noise bandwidth 0.5 Hz, 30.0 Hz, respectively. The driving noise std. was always 0.01 m/s which mismatches the motion model, but still filters out. The FG-based filter ran with three iterations.

12.2 Results

In Fig. 12, we plot the results obtained by the WNav receiver. The left column depicts TPEs and the right col-

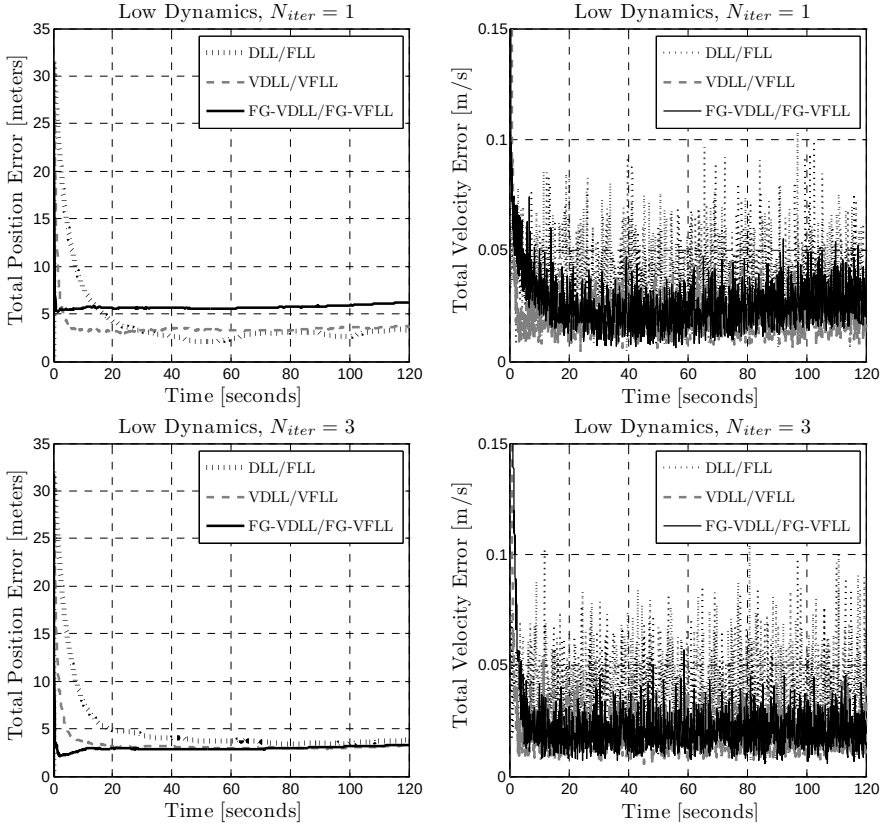

Fig. 9. Vector tracking - simulation results - LD. (a - upper left) TPE, $N_{\text{Iter.}} = 1$, (b - upper right) TVE, $N_{\text{Iter.}} = 1$, (c - lower left) TPE, $N_{\text{Iter.}} = 3$, (d - lower right) TVE, $N_{\text{Iter.}} = 3$.

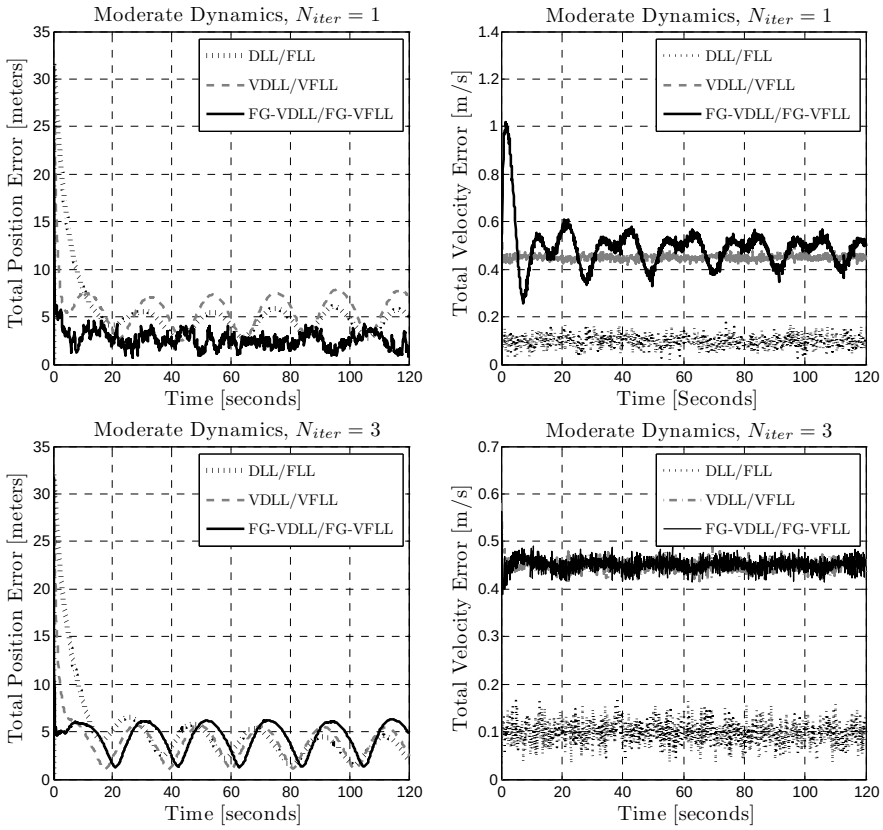

Fig. 10. Vector tracking - simulation results - MD. (a - upper left) TPE, $N_{\text{Iter.}} = 1$, (b - upper right) TVE, $N_{\text{Iter.}} = 1$, (c - lower left) TPE, $N_{\text{Iter.}} = 3$, (d - lower right) TVE, $N_{\text{Iter.}} = 3$.

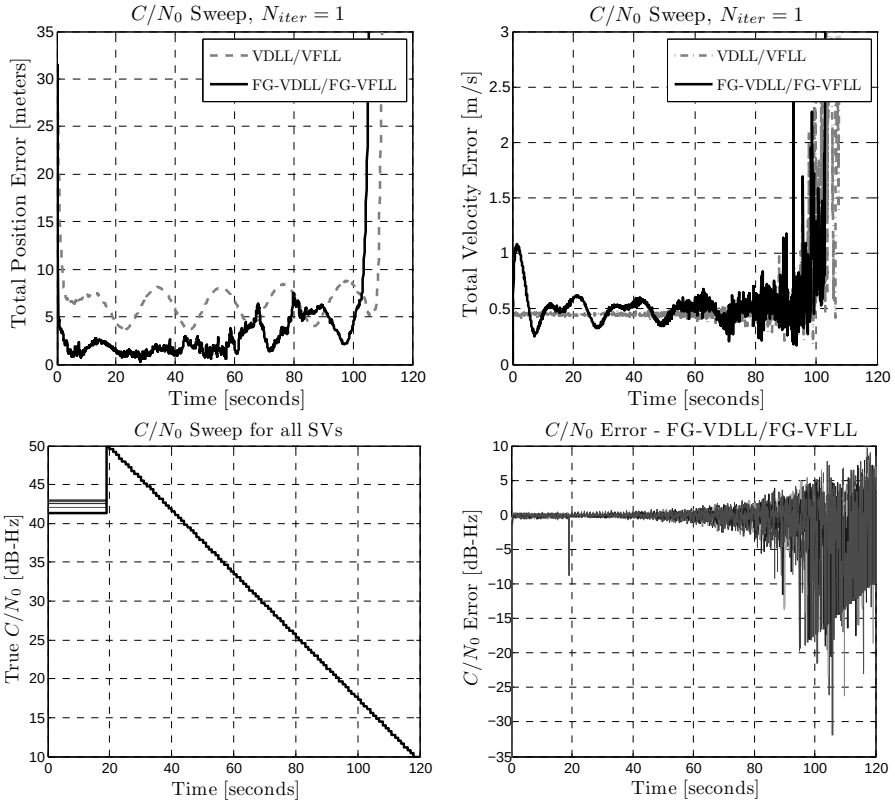

Fig. 11. Vector tracking - simulation results - C/N_0 sweep**.** (a - upper left) TPE at C/N_0 sweep, (b - upper right) TVE at C/N_0 sweep, (c - lower left) swept C/N_0 of all visible SVs, (d - lower right) C/N_0 estimate error for all visible SVs. Figures (a-d) refer to MD, $N_{\text{Iter.}} = 1$.

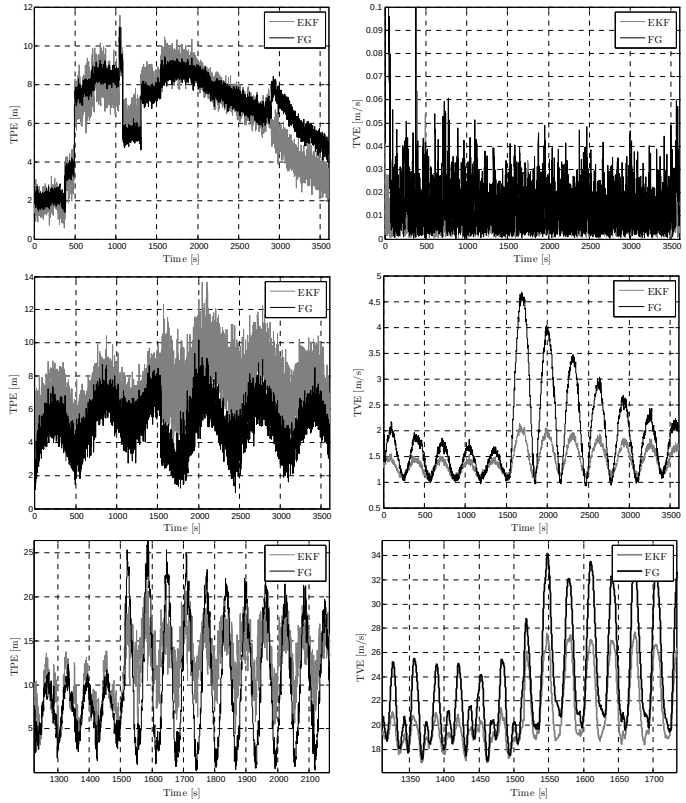

Fig. 12. Experimental results - accuracy. Total position errors (TPE) are in the left column, total velocity errors (TVE) are in the right column. (a - upper left), (b - upper right) averaged errors of static scenarios, (c - middle left), (d - middle right) dynamic scenario $a = 1\,\text{m/s}^2$, (e - lower left), (f - lower right) detail on dynamic scenario $a = 10\,\text{m/s}^2$.

umn depicts TVEs. Figs. 12(a-b) are averaged versions of the TPEs and TVEs for the FG-based and the EKF filters applied to the static user. The accuracy of both filters is comparable. However, convergence was observed only in 10/14 cases. The reason is as discussed in Section 10 - low number of the SVs causes divergence.

In Figs. 12(c-d), the user moves with acceleration $a = 1$ m/s^2, in Figs. 12(e-f) with acceleration $a = 10$ m/s^2 which is not far from the model of the filter. The important fact is that the FG-based filter remains stable and its accuracy is comparable to the EKF's.

13. Conclusion

In this paper, a novel PVT estimation/filtering method for GNSS receiver has been introduced. The method iteratively and distibutively estimates the PVT in a factor graph where each update is calculated using only simple arithmetic operations, thus opening possibilities for implementation in hardware logic.

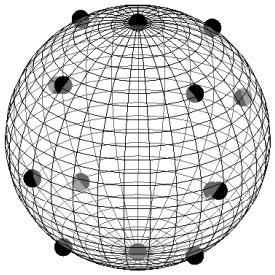

Fig. 13. User positions on the Earth - static experiment (14 positions).

It was demonstrated by Monte Carlo simulations that the method converges for the number of tracked satellites larger than 16, and can withstand higher unmodeled dynamics by increasing the number of iterations. The accuracy of the method is comparable to the accuracy of the EKF at three iterations. At higher dynamics, the proposed method can outperform the EKF if the number of iterations is increased. The method experience comparable number of flops as the EKF in the information form.

It was shown by a case simulation that the proposed method can be coupled with the tracking loops in the vector tracking loop as the EKF and deliver similar performance. By implementation of the method into a real-time receiver developed at CTU in Prague, we showed in several experiments that the method has the potential to withstand channel anomalies.

Acknowledgements

The authors would like to thank JAN SYKORA for his invaluable suggestions and comments.

SafeLOC is a project that is supported by public funds under Contract No. 20110198 with the Technology Agency of the Czech Republic in the program to support research and development ALFA.

References

[1] PANY, T. *Navigation Signal Processing for GNSS Software Receiver - GNSS Technology and Applications.* 1st ed. Norwood: Artech House, 2010.

[2] STOBER, C. et al. ipexSR: A real-time multi-frequency software GNSS receiver. In *Proceedings of ELMAR.* Zadar (Croatia), 2010, p. 407 - 416.

[3] JAKUBOV, O., KOVAR, P., KACMARIK, P., VEJRAZKA, F. The Witch Navigator - a low cost GNSS software receiver for advanced processing techniques. *Radioengineering*, 2010, vol. 19, no. 4, p. 536 - 543.

[4] ABASSIAN, N., PETOVELLO, M. G. Implementation of dual-frequency GLONASS and GPS L1 C/A software receiver. *The Journal of Navigation*, 2010, no. 63, p. 269 - 287.

[5] AFZAL, H., LACHAPELLE, G. Design methodology for a dual frequency configurable GPS receiver. In *Proceedings of GNSS10.* Portland (Oregon), 2010, p. 1 - 9.

[6] SHIVARAMAIAH, N. C., DEMPSTER, A. G. *Global Navigation Satellite Systems - Signal, Theory and Applications: Chapter 2 Baseband Hardware Design in Modernised GNSS Receivers.* 1st ed. InteChOpen, 2012.

[7] KAPLAN, E. D., HEGARTY, C. J. *Understanding GPS: Principles and Applications.* 2nd ed. Norwood: Artech House, 2006.

[8] WAN, E., VAN DER MERWE, R. The unscented Kalman filter for nonlinear estimation. In *Proceedings of IEEE Symposium on Adaptive Systems for Signal Processing, Communications and Control.* 2000, p. 604 - 611.

[9] BO, T., PINGYUAN, C., YANGZHOU, C. Sigma-point Kalman filters for GPS based position estimation. In *Proceedings of Fifth International Conference on Information, Communications and Signal Processing.* 2005, p. 213 - 217.

[10] DOUCET, A., FREITAS, N., GORDON, N. *Sequential Monte Carlo Methods in Practice.* 1st ed. New York: Springer, 2001.

[11] ARULAMPALAM M. S., MASKELL, S., GORDON, N., CLAPP, T. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Transaction on Signal Processing*, 2002, vol. 50, no. 2, p. 174 - 188.

[12] GUSTAFSSON, F. et. al. Particle filters for positioning, navigation, and tracking. *IEEE Transactions on Signal Processing*, 2002, vol. 50, no. 2, p. 425 - 437.

[13] SPILKER, J. J. *Vector delay lock loop processing of radiolocation transmitter signals (US Patent 5398034).* [Online] Cited 2011-10-11. Available at: http://www.freepatentsonline.com/5398034.html.

[14] PETOVELLO, M. G., LACHAPELLE, G. GNSS solutions: What are vector tracking loops and what are their benefits and drawbacks. *Inside GNSS*, 2009, vol. 4, no. 3, p. 16 - 21. [Online] Available at: http://insidegnss.com/node/1458.

[15] PANY, T., EISSFELLER, B. Use of vector delay lock loop receiver for GNSS signal power analysis in bad signal conditions. In *Proceedings of IEEE/ION PLANS 2006.* San Diego (CA, USA), 2006, p. 893 - 903.

[16] KSCHISCHANG, F. R., FREY, B. J., LOELIGER, H. A. Factor graphs and the sum-product algorithm. *IEEE Transaction on Information Theory*, 2004, vol. 47, no. 2, p. 498 - 519.

[17] LOELIGER, H. A. An introduction to factor graphs. *IEEE Signal Processing Magazine*, 2004, vol. 21, no. 1, p. 28 - 41.

[18] LOELIGER, H. A. Least squares and Kalman filtering on Forney graphs. BLAHUT, R. E., KOETTER, R. (Eds.) *Codes, Graphs, and Systems*. Kluwer, 2002, p. 113 - 135.

[19] CHEN, J. C., MAA, C. S., WANG, Y. C. Mobile position location using factor graphs. *IEEE Communications Letters*, 2003, vol. 7, no. 9, p. 431 - 433.

[20] CHEN, J. C., WANG, Y. C., MAA, C. S., CHEN, J. T. Network-side mobile position location using factor graphs. *IEEE Transaction on Wireless Communications*, 2006, vol. 5, no. 10, p. 2696 - 2704.

[21] SPILKER, J. J. Fundamentals of signal tracking theory. PARKINSON, B. W., SPILKER, J. J. (Eds.) *Global Positioning System: Theory and Applications*, 1st ed. Washington DC (USA): American Institute of Aeronautics and Astronautics, Inc., 1996, p. 245 - 327.

[22] BENSON, D. Interference benefits of a vector delay lock loop (VDLL) GPS receiver. In *Proceedings of the 63rd Annual Meeting of the Institute of Navigation.* Cambridge (MA, USA), 2007, p. 1 - 8.

[23] CLOSAS, P., FERNANDEZ-PRADES, C., BERNAL, D., FERNANDEZ-RUBIO, J. A. Bayesian direct position estimation. In *Proceedings of ION GNSS 21st International Technical Meeting of the Satellite Division.* Savannah (GA, USA), 2008, p. 183 - 190.

[24] CLOSAS, P., FERNANDEZ-PRADES, C., FERNANDEZ-RUBIO, J. A. A Bayesian approach to multipath mitigation in GNSS receivers. *IEEE Journal of Selected Topics in Signal Processing*, 2009, vol. 3, no. 4, p. 1 - 12.

[25] CLOSAS, P. *Bayesian Signal Processing Techniques for GNSS Receivers: From Multipath Mitigation to Positioning.* PhD thesis. Barcelona (Spain): Universitat Politecnica de Catalunya, 2009.

[26] KAY, S. M. *Fundamentals of Statistical Signal Processing: Estimation Theory.* 1st ed. New Jersey (USA): Prentice-Hall, 1993.

[27] WATKINS, D. S. *Fundamentals of Matrix Computations.* 2nd ed. New York (USA): John Wiley & Sons, 2002.

[28] KOVAR, P., KACMARIK, P., VEJRAZKA, F. High performance Galileo E5 correlator design. In *Proceedings of 13th IAIN World Congress.* Bergen (Sweden), 2009, p. 1 - 8.

[29] BORIO, D., ANANTHARAMU, P. B., LACHAPELLE, G. SATL-Sim: A semi-analytic framework for fast GNSS tracking loop simulations. *GPS Solutions*, 2011, p. 1 - 5.

[30] JAKUBOV, O., KACMARIK, P., KOVAR, P., VEJRAZKA, F. Universality and realistic extensions to the semi-analytic simulation principle in GNSS signal processing. *Radioengineering*, 2012, vol. 21, no. 2, p. 536 - 543.

About Authors...

Ondrej JAKUBOV was born in Liberec, Czech republic, in 1986. He received his MSc in electrical engineering from the CTU in Prague in 2010. He is a postgraduate student at the same university at the Department of Radio Engineering and works as a navigation engineer for Nottingham Scientific Limited. His research interests include GNSS signal processing algorithms and receiver architectures. His research focuses on advanced positioning and tracking methods. In his occupation, he works as a DSP hardware logic engineer and system architect of a system-on-chip based GNSS receiver, provides support and co-developes RF front-end solutions.

Pavel KOVAR was born in Uherske Hradiste in 1970. He received his MSc and PhD degree at the CTU in Prague in 1994 and 1998. Since 1997 to 2000 he worked in Mesit Instruments as a designer of avionics instruments. Since 2000 he is with the Faculty of Electrical Engineering of the CTU in Prague as an Assistant Professor and since 2007 as an Associate Proffesor (Doc.). His interests are receiver architectures, satellite navigation, digital communications and signal processing.

Petr KACMARIK was born in 1978. He received his MSc degree from the CTU in Prague in 2002. During his postgraduate studies (2002 - 2006) he was engaged in the investigation of satellite navigation receiver techniques in hard conditions and wrote his PhD thesis with its main topic on GNSS signal tracking using DLL/PLL with applicability to week signal environment. He defended the thesis in 2009. Since 2006 he works as an Assistant Professor at the Faculty of Electrical Engineering of the CTU in Prague. His main professional interests are satellite navigation, digital signal processing, implementations and simulations of signal processing algorithms.

Frantisek VEJRAZKA was born in 1942. He received his MSc degree from the CTU in Prague in 1965. He served as an Assistant Professor (1970) and Associate Professor (1981) at the Department of Radio Engineering. He is a full professor of radio navigation, radio communications, and signals and systems theory since 1996. He was appointed the Head of the Department of Radio Engineering (1994–2006), vice-dean of the Faculty (2000–2001) and vice-rector of the CTU in Prague (2001–2010). His main (professional) interest is in radio satellite navigation where he participated on the design of the first Czech GPS receiver (1990) for the Mesit Instruments. He was responsible for the development of Galileo E5 receiver for Korean ETRI during 2008 and E1 and E5a receiver in 2009. He is the former president of the Czech Institute of Navigation, Fellow of the Royal Institute of Navigation in London, member of the Institute of Navigation (USA), vice-president of IAIN, vice-chairman of CGIC/IISC, member of IEEE, member of Editorial Board of GPS World, Editorial Board of InsideGNSS, etc.